PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-031599
(43)Date of publication of application : 03.02.1998

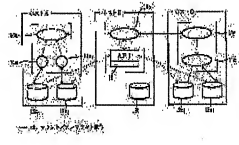
(51)Int.Cl. G06F 12/00
G06F 12/00

(21)Application number : 08-184383
(22)Date of filing : 15.07.1996

(71)Applicant : NIPPON STEEL CORP
(72)Inventor : NITTOME TERUHIDE
KAWACHI HISAKAZU

(54) DATA STORAGE SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To easily create an application software for handling a large amount of data excessing a file capacity by making access to a file on a hard disk of other host computer on a network and to easily handle the data in the application software for handling a large amount of data excessing capacity of a hard disk.
SOLUTION: Data pool access servers 11a to 11c are provided on plural host computers and an application software 13 makes access to a logic file managed by the respective data pool access servers 11a to 11c on a data pool via the data pool access servers 11a to 11c. The application software 13 writes in and reads out the logic file as a unit. The logic file is recognized as so called a virtual file, observing from the application software 13.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開平10－31599
(43)公開日　平成10年(1998)２月３日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ　12/00	５０１		Ｇ０６Ｆ　12/00	５０１Ａ	
	５４５			５４５Ａ	

審査請求　未請求　請求項の数４　ＯＬ　（全６頁）

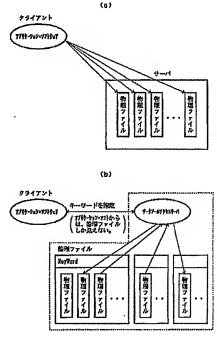
(21)出願番号　特願平8－184383

(22)出願日　平成８年(1996)７月15日

(71)出願人　000006655
新日本製鐵株式会社
東京都千代田区大手町２丁目６番３号
(72)発明者　新留　順英
東京都千代田区大手町２丁目６番３号　新日本製鐵株式会社内
(72)発明者　河内　久和
東京都千代田区大手町２丁目６番３号　新日本製鐵株式会社内
(74)代理人　弁理士　半田　昌男

(54)【発明の名称】　データ記憶システム

(57)【要約】
【課題】　ネットワーク上で他のホストのハードディスク上にあるファイルにアクセスして、ファイル容量の制限を超えるような大量のデータを扱うアプリケーション・ソフトウェアの作成を容易にするとともに、ハードディスクの容量を超えるような大量のデータを扱うアプリケーション・ソフトウェアにおける取扱いを容易にする。
【解決手段】　複数のホスト上にデータプールアクセスサーバを設け、アプリケーション・ソフトウェアは、このデータプールアクセスサーバを介してそれぞれのデータプールアクセスサーバが管理するデータプール上の論理ファイルにアクセスする。アプリケーション・ソフトウェアは、論理ファイルを単位として書き込み又は読み出しを行う。アプリケーション・ソフトウェアから見た場合には、論理ファイルがいわば仮想的なファイルとして認識される。



1

【特許請求の範囲】
【請求項1】　データの読み出し及び書き込みを可能とする記憶媒体と、
前記記憶媒体に対し物理ファイルを単位としてデータの読み出し及び書き込みの動作を制御するオペレーティングシステムと、
データの格納場所であるデータプールを、それが存在するディレクトリ、それを構成する前記物理ファイルの容量及びそのデータプールの容量によって定義するデータプール構成ファイルと、
アプリケーション・ソフトウェアからの要求に従って、一つのデータプール内で、前記論理ファイルを単位として、前記オペレーティングシステムに前記記録媒体に対するデータの記録及び読み出しを行わせるデータプールアクセスサーバと、
を具備することを特徴とするデータ記憶システム。
【請求項2】　前記データプールアクセスサーバは、データの容量が前記物理ファイルの容量を超える場合に、前記データを分割して複数の物理ファイルとして前記データプールに書き込むものである請求項1記載のデータ記憶システム。
【請求項3】　前記データプールアクセスサーバは、データプールにデータの書き込みを行っているときに、そのデータプールの空き容量がなくなったときは、そのデータプール内の論理ファイルのステータスを参照し、既に読み出し済んだものがあれば、それらを削除してデータ書き込み領域を確保する機能を有する請求項1又は2記載のデータ記憶システム。
【請求項4】　前記オペレーティングシステムはＵＮＩＸであり、前記記憶媒体はハードディスクであることを特徴とする請求項1，2又は3記載のデータ記憶システム。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、主としてネットワーク上の他のホストの記憶装置に対してデータの書き込み及び読み出しをする場合に利用されるデータ記憶システムに関する。
【０００２】
【従来の技術】ワークステーション用のオペレーティングシステムとして最も代表的なＵＮＩＸでは、一つの物理ファイルの容量が２ＧＢを超えることができないという制限があり、扱うデータの容量がこの制限を超える場合には、データを複数のファイルに分割するなどの特別な処理を行わなければ、このような大きな容量のデータを扱うことはできない。大量のデータを処理するアプリケーション・ソフトウェア（以下、「プロセス」ともいう）をＵＮＩＸ上で動作させるためには、そのアプリケーション・ソフトウェアを作成する際に、データの容量が２ＧＢを超える場合にどのような処置を施すかという

2

手続きをプログラム中に記述しておく必要がある。
【０００３】
【発明が解決しようとする課題】しかしながら、多様なアプリケーション・ソフトウェアを作成するそのたびごとに、データの容量がファイル容量の制限を超える場合にどのような取扱いとするかという複雑な処理を一々プログラミングするのは、プログラマにとって大きな負担となる。
【０００４】また、複数のホストが接続されたネットワーク上では、通常ＮＦＳ（NetworkFile System）が利用されることが多いが、これを用いてあるホストから別のホストのハードディスクに存在するファイルにアクセスする場合には、マウント／アンマウント操作を行う必要があり、その際、ホスト名、ファイルの所在場所（ディレクトリ名）、ファイル名を意識して使用しなければならない。したがって、アプリケーション・ソフトウェアを作成する場合には、これらの点を考慮してプログラミングする必要があり、プログラミング作業が複雑化する。
【０００５】更に、ハードディスクの容量を超えるような大きなデータを扱う場合には、アプリケーション・ソフトウェアにおいて、サイクリックファイル等の特別な機能を設ける必要がある。この場合、一つのファイルに対して複数のアプリケーション・ソフトウェアがアクセスする場合には、そのファイルにアクセスするすべてのプロセスについてこのような機能を取り入れる必要があり、各プロセスをプログラミングする際にいわゆる排他制御等が複雑となる。
【０００６】本発明は、上記事情に基づいてなされたものであり、特に、ネットワーク上で他のホストのハードディスク上にあるファイルにアクセスして、ファイル容量の制限を超えるような大量のデータを扱うアプリケーション・ソフトウェアの作成を容易にするとともに、ハードディスクの容量を超えるような大量のデータを扱うアプリケーション・ソフトウェアにおける取扱いを容易にするデータ記憶システムを提供することを目的とする。
【０００７】
【課題を解決するための手段】上記の課題を解決するための本発明は、データの読み出し及び書き込みを可能とする記憶媒体と、前記記憶媒体に対し物理ファイルを単位としてデータの読み出し及び書き込みの動作を制御するオペレーティングシステムと、データの格納場所であるデータプールを、それが存在するディレクトリ上の位置、それを構成する前記物理ファイルの容量及びそのデータプールの容量によって定義するデータプール構成ファイルと、アプリケーション・ソフトウェアからの要求に従って、一つのデータプール内で、前記論理ファイルを単位として、前記オペレーティングシステムに前記記録媒体に対するデータの記録及び読み出しを行わせるデ

ータプールアクセスサーバとを具備することを特徴とする。

【０００８】本発明は、上記より、記憶媒体に書き込みを行うデータが、そのオペレーティングシステムの物理ファイルの最大容量（ＵＮＩＸであれば、２ＧＢ）を超えるものであっても、アプリケーション・ソフトウェアの側では、論理ファイルを単位としてデータの書き込み及び読み出しを行うことができるので、例えば２ＧＢごとにデータの分割等の作業を行う必要がない。

【０００９】

【発明の実施の形態】以下に図面を参照して、本発明の一実施形態について説明する。図１は、従来のデータ記憶システムと本実施形態のデータ記憶システムの概念を比較して示した図であり、同図（ａ）が従来のデータ記憶システム、同図（ｂ）が本実施形態のデータ記憶システムに対応する。

【００１０】従来のデータ記憶システムでは、図１（ａ）に示すように、あるホスト上のアプリケーション・ソフトウェアが他のホスト上のファイル（例えばＵＮＩＸのファイルで、物理ファイルという。）にアクセスする場合は、そのアプリケーション・ソフトウェアがあるホストをクライアント、当該ファイルがあるホストをサーバとして、アプリケーション・ソフトウェアから通信ネットワークを介して他のホスト上のファイルにアクセスしていた。また、あるホストのディスク上にあるファイルにアクセスするために、そのディスクのマウント操作を行うと、以後、特別にアンマウント操作を行うまでは、そのディスクはマウントされた状態が続く。

【００１１】これに対して、本実施形態のデータ記憶システムでは、図１（ｂ）に示すように複数のホスト上にソフトウェアであるデータプールアクセスサーバを設け、アプリケーション・ソフトウェアは、このデータプールアクセスサーバを介してそれぞれのデータプールアクセスサーバが管理するデータプール上の論理ファイルにアクセスする。データプールアクセスサーバは、一つのデータプールを管理する。データプールアクセスサーバは、アプリケーション・ソフトウェアからサービスのオープン要求があったときに起動し、一度起動すると、サービスのクローズ要求があるまで常駐する。また、データプールアクセスサーバは、クライアントのアプリケーション・ソフトウェアからサービスのオープン要求があったときのみ関連するディスクをマウントし、サービスのクローズ要求があるとそのディスクのアンマウント操作を行う。このため、不要なときまでマウント状態が持続することが回避され、コンピュータにかかる負荷が軽減される。

【００１２】図１（ｂ）において、物理ファイルは、オペレーティングシステムが管理する一つ一つのファイルであり、ＵＮＩＸの場合であれば、その容量は最大で２ＧＢである。従来は、この最大２ＧＢのファイルを単位として、マウントした他のホストのディスク上にデータの書き込み又は読み出しを行う必要があった。すなわち、クライアント側のアプリケーション・ソフトウェアは、それ自身でアクセスを希望するファイルのオープン、クローズの作業を行なわなければならず、また、データの書き込みを行っている途中に２ＧＢを超えることとなった場合には、アプリケーション・ソフトウェア自身がそのデータを適当な容量に分割し、別の新しいファイルを作って残りのデータを書き込むという作業を行う必要があった。したがって、アプリケーション・ソフトウェアをプログラミングする際には、これらの点を考慮してプログラムを作成する必要があった。

【００１３】これに対し本実施形態では、アプリケーション・ソフトウェアは、論理ファイルを単位として他のホストのハードディスクに対して書き込み又は読み出しを行う。ここで、論理ファイルとは、一つ又は複数の物理ファイルの集合であり、その容量は、１物理ファイルの容量を単位として、それが属するデータプール全体の容量まで変化することができる。データプールアクセスサーバは個々の物理ファイルを管理しており、他のホストのアプリケーション・ソフトウェアからそのデータプールアクセスサーバが管理する論理ファイルへのアクセスを依頼された場合には、その論理ファイルを構成する個々の物理ファイルに対する書き込み・読み出しを実行する。

【００１４】したがって、アプリケーション・ソフトウェアから見た場合には、論理ファイルがいわば仮想的なファイルとして認識される。これにより、容量が２ＧＢを超えるデータであっても、その場合のデータの分割等の処理はデータプールアクセスサーバが自動的にオペレーティングシステムを介して行ってくれるので、アプリケーション・ソフトウェアの側ではそのことを意識する必要がなく、アプリケーション・ソフトウェアが行わなければならない作業が簡素化される。このため、アプリケーション・ソフトウェアのプログラミング作業が簡単になり、プログラマの負担が軽減され、ひいてはアプリケーション・ソフトウェアの開発コストが低減する。

【００１５】論理ファイルには、「空」、「書き込み中」、「書き込み済」、「読み出し中」、「読み出し済」という五つのステータスを与える。「空」は、その論理ファイルにデータが含まれていない状態を示す。「書き込み中」は、その論理ファイルがディスク上にデータを書き込んでいる最中であることを示す。「書き込み済」は、ディスク上でその論理ファイルにデータの書き込みが終了したことを示す。「読み出し中」は、その論理ファイルがデータの読み出し中であることを示す。そして「読み出し済」は、その論理ファイルから既にデータの読み出しが終了していることを示す。このようなステータスを与えることによって、同一の論理ファイルに対して複数のアプリケーション・ソフトウェアからア

5　　6

クセスされる際の排他制御が可能となり、また、そのデータが既に読み出されて別の形態で他の場所に送られているかどうかなどを確認することができる。

【0016】図2は、本実施形態のネットワーク接続されたデータ記憶システムの構成を示す図である。同図では3台のホストA，B，Cがネットワークに接続されている例を示す。このうち、ホストBがクライアントマシン、ホストA及びCがサーバマシンである。各ホストには、ソケット通信を行うための通信ソフトウェア10a，10b，10cが設けられている。サーバーマシンであるホストAには、データプールアクセスサーバ11a$_1$，11a$_2$が搭載されており、同じくサーバマシンであるホストCには、データプールアクセスサーバ11cが搭載されている。それぞれのデータプールアクセスサーバは、一つのデータプールを管理している。一つのデータプールには、複数の論理ファイルを書き込むことができる。また、各ホストは、それぞれにハードディスク12a$_1$，12a$_2$，12$_{x1}$，12$_{x2}$を有している。図2では、サーバーマシンであるホストAのデータプールアクセスサーバ11a$_1$はハードディスク12a$_1$をデータプールとして使用し、データプールアクセスサーバ11a$_2$はハードディスク12a$_2$をデータプールとして使用する。また、同じくサーバーマシンであるホストCのデータプールアクセスサーバ11a$_1$はハードディスク12$_{z1}$及び12$_{z2}$をデータプールとして使用する。

【0017】一方、クライアントマシンであるホストBには、現在ホストB上で稼働しているアプリケーション・ソフトウェアI3が搭載されている。なお、ホストBの中に「API」として示したものは、必要に応じてアプリケーションプログラムの中に記載し、アプリケーションプログラムの実行時に呼び出されるライブラリ関数を概念的に示したものである（以下「API関数」という）。アプリケーション・ソフトウェアは、このAPI関数を介して、本実施形態のデータ記憶システムの種々の機能を利用する。このAPI関数には、サービスの利用開始を宣言する関数、サービスの利用終了を宣言する関数、論理ファイルをオープンしてデータの入出力を可能にする関数、論理ファイルをクローズしてデータの入出力を終了させる関数などが含まれる。

【0018】尚、図2において、点線の矢印は、アプリケーション・ソフトウェアによるサーバーのハードディスク12a$_1$，12a$_2$，12$_{z1}$，12$_{z2}$に対するマウント／アンマウント操作を示す。次に、本発明のデータプール構成ファイルであるコンフィギュレーション・ファイルについて説明する。データプールの所在、容量等の管理情報は、このコンフィギュレーション・ファイルに記述される。図3は、このコンフィギュレーション・ファイルの一例を示した図である。同図において、「Name」の欄にはデータプールの識別名を記述し、「Host」の欄にはデータプールが存在するホストのホスト名を記述し、「Directory」の欄にはデータプールが存在するディレクトリを記述し、「File Size」の欄には1物理ファイルの容量をバイト単位で記述し、「Pool Size」の欄には一つのデータプールの容量をキロバイト単位で記述する。このファイルは、予めユーザーが記述して、各ホストに分配しておく。

【0019】図3に示したコンフィギュレーション・ファイルでは、pool1、pool2、pool3という三つのデータプールが定義されている。図3のコンフィギュレーション・ファイルによれば、「pool1」というデータプールは、「hostA」というホストに存在し、その容量は2GBで、これを構成するそれぞれの物理ファイルの大きさは20MBである。また、「pool2」というデータプールは、「hostA」というホストに存在し、その容量は10GBで、これを構成するそれぞれの物理ファイルの大きさは10MBである。更に、「pool3」というデータプールは、「hostC」というホストに存在し、その容量は4GBで、これを構成するそれぞれの物理ファイルの大きさは10MBである。

【0020】本実施形態のデータ記憶システムは、このコンフィギュレーション・ファイルに従ってデータプールのファイル構成を定義する。アプリケーション・ソフトウェアは、コンフィギュレーション・ファイルに記述されたデータプールの識別名のみを認識し、API関数を呼び出すだけで、ネットワークでつながれているホスト上にあるデータプール内の任意の論理ファイルを操作することが可能となる。このとき、アプリケーション・ソフトウェアは、その論理ファイルを構成する個々の物理ファイル、すなわちOSが扱うファイルを意識する必要はない。

【0021】論理ファイルをデータプールに書き込む場合、その論理ファイルには必ずキーワードを与える。このキーワードが、そのデータプール内でその論理ファイルを特定する手段となる。アプリケーション・ソフトウェアは、データプール識別名とキーワードを与えることによって、所望の論理ファイルを獲得し、以後、この論理ファイルに対して書き込み及び読み出しを行うことができる。同一のデータプール内に複数の論理ファイルを書き込む場合には、このキーワードによって各論理ファイルを識別することができる。また、このようなキーワードを付けることにより、データを読み出すときに、キーワードによって論理ファイルの検索を行うことができる。このとき、キーワードとしてその論理ファイルを作成した日付に基づいた数字列を用いる場合を考え、「>」、「<」、「=」といった数学記号を用いた検索を可能とする。

【0022】データプールに対してある論理ファイルの

データを書き込んでいる途中にそのデータプールの容量が一杯となり、更にその論理ファイルの残りのデータを書き込む必要がある場合には、そのデータプールに既に書き込まれているすべての論理ファイルのステータスを見て、既に読み込み済となっている論理ファイルのうちで最も古いものを削除して新たなデータスペースを確保し、そこにデータを書き込む。このとき、仮に、読み込み済のものがない場合には、

1．読み込む論理ファイルができるまで待機する

2．ファイルを獲得できない旨のリターンコードを即リターンする

のいずれかの動作をするように、アプリケーション・ソフトウェアの側で選択できるようにする。

【００２３】尚、本発明は、上記実施形態に限定されるものではなく、その要旨の範囲内で種々の変更が可能である。例えば、上記実施形態では、論理ファイルに５種類のステータスを与えたが、この中の「読み込み済」のステータスについて、何回読み込みが行われたかをカウントし、一定回数の読み込みが終了したら、古い順から削除可能とするような構成とすることもできる。また、上記実施形態では、ネットワークに接続されたクライアント側のアプリケーション・ソフトウェアが、サーバ上のディスクに論理ファイルの書き込みや読み出しを行う場合について説明したが、本発明は、例えばＵＮＩＸを搭載した１台のコンピュータ上でも動作させることが可能である。

【００２４】

【発明の効果】以上説明したように、本発明によれば、＊データの容量が、そのオペレーティングシステムの物理ファイルの最大容量を超える場合であっても、データの分割等の処理はデータプールアクセスサーバが自動的にオペレーティングシステムを介して行ってくれるので、アプリケーション・ソフトウェアの側ではそのことを意識する必要がなく、単に論理ファイルだけを認識してデータの書き込み及び読み出しを行えばよいので、アプリケーション・ソフトウェアが行わなければならない作業が簡素化され、アプリケーション・ソフトウェアを開発する際のプログラミング作業が簡素化されてプログラマの負担が軽減され、ひいてはアプリケーション・ソフトウェアの開発コストが低減するデータ記憶システムを提供することができる。

【図面の簡単な説明】

【図１】従来のデータ記憶システムと本実施形態のデータ記憶システムの概念を比較して示した図である。

【図２】本実施形態のネットワーク接続されたデータ記憶システムの構成を示す図である。

【図３】コンフィギュレーション・ファイルの一例を示した図である。

【符号の説明】

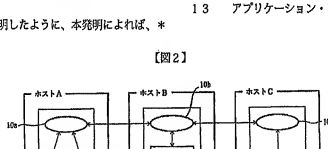
１０ａ，１０ｂ，１０ｃ　通信ソフトウェア

$11a_1$，$11a_2$，１１ｃ　データプールアクセスサーバ

$12a_1$，$12a_2$，１２ｂ，$12c_1$，$12c_2$　ハードディスク

１３　アプリケーション・ソフトウェア

【図２】



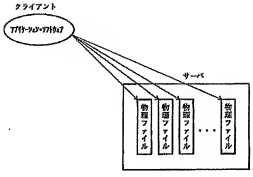
…… は、マウント/アンマウント操作

【図３】

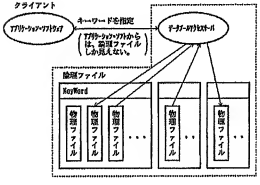
Name	Host	Directory	File Size(Byte)	Pool Size(Byte)
pool1	hostA	/users/pool/pool1	20000000	2000000
pool2	hostA	/users/pool/pool2	10000000	10000000
pool3	hostC	/users/pool/pool3	10000000	4000000

【図１】

(a)


クライアント
アプリケーション・ソフトウェア
サーバ
物理ファイル
物理ファイル
物理ファイル
物理ファイル

(b)


クライアント
アプリケーション・ソフトウェア
キーワードを指定
アプリケーション・ソフトからは、論理ファイルしか見えない。
データ・ストレージ・システム
論理ファイル
KeyWord
物理ファイル
物理ファイル
物理ファイル
物理ファイル
物理ファイル